Flag Slurper Documentation

Release 0.4

Matt Gerst

Nov 03, 2018

Contents

Flag slurper is a Red Team utility for *Cyber Defense Competitions*. It provides Auto PWN functionality, as well as functionality for tracking obtained credentials, files, and most importantly, flags.

CHAPTER 1

Auto PWN

Flag Slurper contains a utility for automatically attempting default credentails against teams' SSH hosts. This works by grabbing the team list from *IScorE* and a list of all the services. The default credentails it uses are:

- `root:cdc`
- `cdc:cdc`

1.1 Requirements

AutoPWN requires a database. For many cases sqlite will do, but in order to use parallel AutoPWN, a server-based database (such as postgres) is required. This is due to sqlite only allowing one writer at a time. The database can be configured in your flagrc file:

```
[database]
; For sqlite (default)
url=sqlite:///{{ project }}/db.sqlite

; For postgres
url=postgres:///splurper
```

The `{{ project }}` variable is the file path to the current project and is optional.

1.2 Usage

You first need to create a project and result database:

```
flag-slurper project init -b ~/cdcs/isu2-18 --name "ISU2 2018"
flag-slurper project create-db
```

To generate the team and service list you can simply run:

```
flag-slurper autopwn generate
```

This will cache the team and service lists into the database. This will be used by other `autopwn` commands so they don't need to keep hitting the *IScorE* API during the attack phase when the API is getting hammered.

After generating the local files, you can then pwn all the things!

```
flag-slurper autopwn pwn
```

This will print out what credentials worked on which machines and any flags found. These results are recorded in the database and can be viewed like this:

```
flag-slurper autopwn results
```

CHAPTER 2

Projects

Flag slurper has the concept of "projects". These projects tell flag slurper where to find various files such as the `teams.yml` and `services.yml` files. It may also contain other configuration options such as where flags are located. The primary purpose of the project system is to keep data from different *CDCs* separate.

To create a project, run:

```
flag-slurper project init --base ~/cdcs/isu2-18 --name "ISU2 2018"
```

This will create a project named "ISU2 2018" in the folder `~/cdcs/isu2-18`. You can then run the following command to activiate the project.

```
eval $(flag-slurper project env ~/cdcs/isu2-18)
```

Note: The output of the `env` command will set the `SLURPER_PROJECT` environment variable. This variable can also be set manually, instead of the `--project` flag.

When you want to deactivate a project, run the `unslurp` command.

Alternatively, you can specify `--project PATH` on each command. For example:

```
flag-slurper --project ~/cdcs/isu2-18/ autopwn generate
```

Note: The `--project PATH` flag must be before any subcommands.

2.1 Flags

The Auto PWN feature will automatically look in common directories for *flags* that look like a flag. You can also specify locations to check. The following project file defines the "Web /root flag":

```
_version: "1.0"
project: ISU2 2018
base: ~/cdcs/isu2-18
flags:
  - service: WWW SSH
    type: blue
    location: /root
    name: "team{{ num }}_www_root.flag"
    search: yes
```

You can specify as many flags as you want. All of the following fields are required:

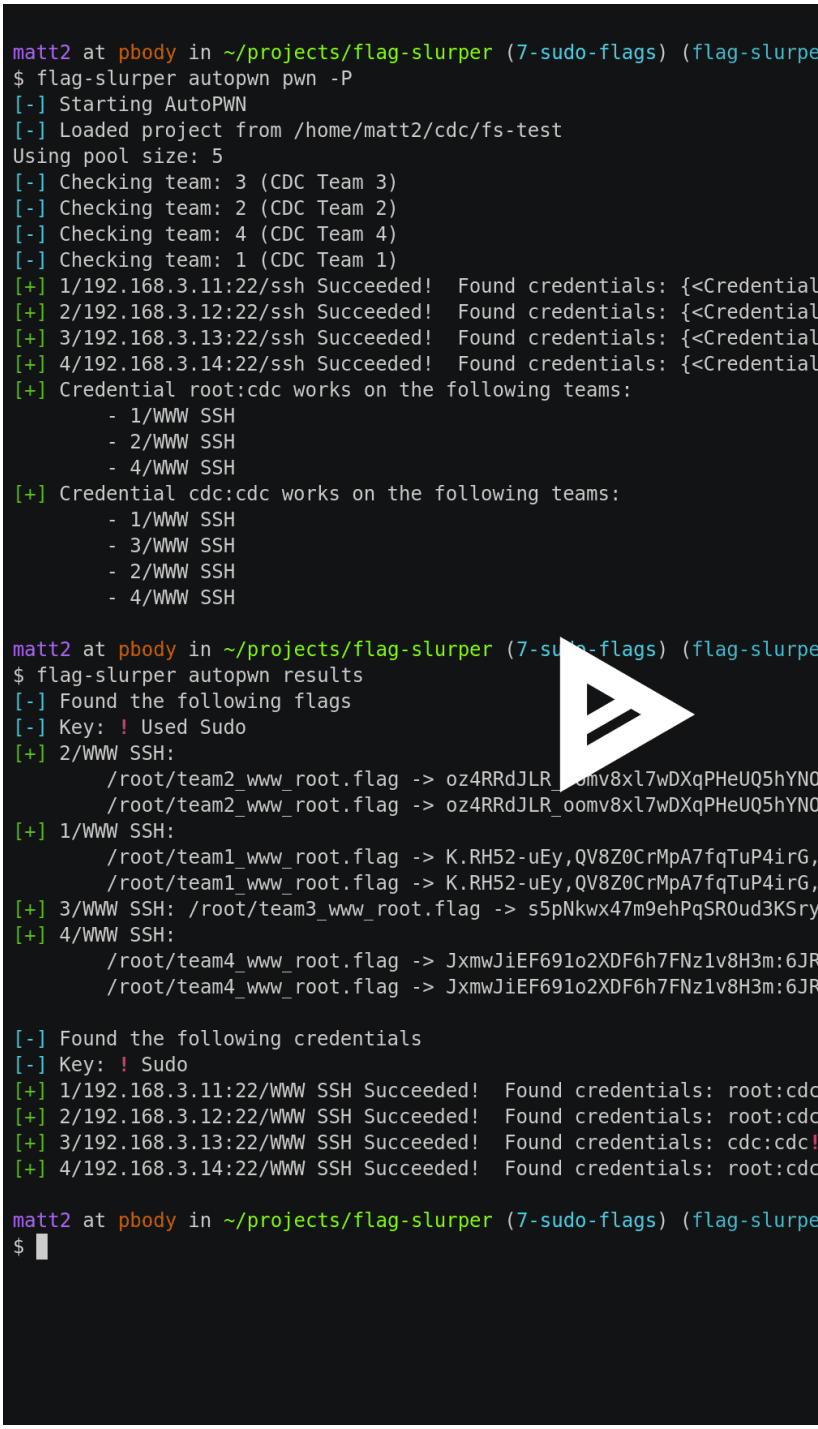
service The name of the service this flag is associated with. Auto PWN matches against this when determining what flags it should look for when attacking a service.

type Which flag type this is `blue` (read) or `red` (write). Currently only `blue` is supported.

location The directory the flag is supposed to be located in.

name The expected file name of the flag. Pay close attention to `{{ num }}`. This is a placeholder that will be replaced with the team number during the attack.

search Whether Auto PWN should search `location` for any files that are roughly the correct file size. A search is only performed if the falg is not found at it's exact name `{{ location }}/{{ name }}`.



Here's an example of an Auto PWN run that obtained flags:

2.2 Credentials

Credentails can be managed through the `creds` subcommand. To add a credential:

```
flag-slurper creds add root cdc
```

List credentials:

```
flag-slurper creds ls
```

Remove credential:

```
flag-slurper creds rm root cdc
```

Show details for a credential

```
flag-slurper creds show root:cdc
```

CHAPTER 3

Glossary

IScorE IScorE is the scoring system built and used by ISEAGE during their *CDCs*.

CDC Cyber Defense Competition.

Flag A file on the teams' system representing sensitive data. Red team's goal is to place red flags, and to read blue flags placed on the system by the Blue Teams.

Red Team The attacking team.

Blue Teams The defending teams.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Index